OPTIMAL PROCESSOR SCHEDULING WHEN SERVICE TIMES ARE HYPEREXPONENTIALLY DISTRIBUTED AND PREEMPTION OVERHEAD IS NOT NEGLIGIBLE

by

Kenneth C. Sevcik

Technical Report CSRG - 16
June 1972

COMPUTER SYSTEMS RESEARCH GROUP
UNIVERSITY OF TORONTO

OPTIMAL PROCESSOR SCHEDULING WHEN SERVICE TIMES ARE HYPEREXPONENTIALLY DISTRIBUTED AND PREEMPTION OVERHEAD IS NOT NEGLIGIBLE

by

Kenneth C. Sevcik

Technical Report CSRG - 16
June 1972

The Computer Systems Research Group (CSRG) is an interdisciplinary group formed to conduct research and development relevant to computer systems and their application. It is jointly administered by the Department of Electrical Engineering and the Department of Computer Science of the University of Toronto, and is supported in part by the National Research Council of Canada.

ABSTRACT

When scheduling requests with hyperexponentially distributed service times on a single processor, preemption may be required in order to minimize the average time in system. However, if processor time is lost during preemption, the resulting delay of all requests must be balanced against the advantage of preemption. We will develop some characteristics of an optimal preemptive scheduling rule for this situation, and we will compare the performance of an optimal rule with the performance of some well known rules.

I. INTRODUCTION

Consider the problem of minimizing average time in system while serving requests with hyperexponential service times when the overhead of preemption cannot be ignored. We presume that a partially completed request may be interrupted and later continued from the point of interruption (preemptive-resume scheduling), but that the operation of recording the state of the interrupted process requires some amount of processor time. This problem is interesting because empirical investigations have shown that service time distributions in some computer systems are approximately hyperexponential [1].

A hyperexponential distribution is the weighted sum of exponential distributions. Through most of this paper, we use one which is the weighted

sum of two exponential distributions:

$$f(t) = aue^{-ut}+(1-a)ve^{-vt}$$

where $0 \leq a \leq 1$ and (by convention) $u \leq v$.

A hyperexponential distribution has a higher variance than an exponential distribution with the same mean (see Figure 1). If t is exponentially distributed, the quantity $E(t-x|t>x)$ is independent of the conditioning variable, x. This is a consequence of the "memoryless property" [2] of the exponential distribution. However, if t is hyperexponentially distributed, the quantity $E(t-x|t>x)$ increases monotonically with x (unless either $u=v$ or $a(1-a)=0$). Intuitively, this means that when service times are hyperexponentially distributed, the longer a particular request has been served without being completed, the greater the amount of service it is expected to need additionally in order to be completed. In Figure 2, the conditional mean of the hyperexponential distribution is compared with the conditional means of the exponential distributions $f(t) = ue^{-ut}$ (upper dotted curve) and $f(t) = we^{-wt}$ where $w = uv/(av+(1-a)u)$ (lower dotted curve). The curve labelled R(x) will be discussed later.

Several previous results are of relevance to this problem. When service times are hyperexponentially distributed and preemption causes no lost processor time, average time in system is minimized by serving at all times the request with the least attained service time [3]. This scheduling strategy has been called "preemptive processor sharing" (PPS) [4]. It is the limiting case of a multi-level queue with feed back discipline (FB) [5]. Under FB, requests initially enter the highest priority queue. The first request in the highest priority queue which is not empty is assigned service for a quantum associated with that queue level. If it is completed, it

leaves the system, otherwise it joins the end of the next lower level queue, and the scheduling rule is reapplied. As the quantum lengths assigned at each level approach zero, FB approaches PPS.

Another relevant result is that preemption is not needed to minimize average time in system when service times are exponentially distributed [6]. The "smallest rank" scheduling rule (SR) is based on the intuitive goal of minimizing the ratio of expected processor time invested to the expected success in completing a request. The ranking attribute is calculated from the conditional distribution of remaining service time. In Figure 2, the curve R(x) is the current rank of a request with attained service x. It has been proven that SR minimizes average time in system when preemption is free. If service times are hyperexponentially distributed, SR specializes to PPS. If service times are exponentially distributed, the rank of a request is independent of its attained service, and SR allows any scheduling order [3].

We now wish to reconsider these results with the more realistic assumption that whenever a preemption occurs, there is an interval of time, C>0, in which the processor is not providing service to any request. In this case a scheduling rule approaching PPS becomes undesirable because the percentage of time that the processor is providing useful service approaches zero. Thus selecting a scheduling strategy for this situation involves balancing the desirability of running a request with small attained service against the loss of processor time during preemptions.

In the remainder of this paper we will investigate various characteristics of an optimal preemptive scheduling rule. The analytic work will be in terms of the parameters a, u, v, and C. In all numeric examples and

Digitized by the Internet Archive
in 2018 with funding from
University of Toronto

https://archive.org/details/technicalreportc16univ

calculations the parameter values, unless specified otherwise, will be a=1/10, u=1/40, v=1/4, and C=1/8.

II. CONDITIONS WHICH MAKE PREEMPTION UNDESIRABLE

With attained service x, the conditional distribution of service time remaining of the hyperexponential distribution is

$$f(t-x)_{|t>x} = \frac{(aue^{-u(t-x)} + (1-a)ve^{-(vt-ux)})}{a+(1-a)e^{-(v-u)x}} .$$

As x approaches infinity, this becomes

$\lim_{x\to\infty} f(t-x)_{|t>x} = ue^{-u(t-x)}$, which is precisely one of the exponential distributions from which the hyperexponential distribution was formed. Figure 2 also indicates that the properties of the hyperexponential distribution asymptotically approach those of an exponential distribution as x increases.

No matter how small C is, there is a point of attained service at which the conditional distribution is so close to exponential that preemption in favor of a request with that much attained service is never desirable. We can solve for this point as follows. Presume that all n requests currently in system have attained service of at least x. We will determine the smallest such x for which preemption of another request in favor of a request with attained service x is never desirable. Preemption is most desirable if n-1 requests have attained service x while the n^{th} request has a value of attained service which becomes arbitrarily large. Without preemption, the expected time of the i^{th} completion is

$$\lim_{y\to\infty} E[t|t>y] + (i-1)E[t|t>x].$$

With preemption, the expected time of the i^{th} completion is

$$C + (i)E[t|t>x] \quad \text{for } i=1,\ldots,n-1.$$

The expected time of the n^{th} completion is

$$C + (n-1)E[t|t>x] + \lim_{y\to\infty} E[t|t>y].$$

Summing over the n requests, we find that average time in system is <u>not</u> reduced by preemption if

$$nC + (n(n+1)/2 - 1)E[t|t>x] + \lim_{y\to\infty} E[t|t>y] \geq (n) \lim_{y\to\infty} E[t|t>y] + (n(n-1)/2)E[t|t>x],$$

or

$$nC + (n-1)E[t|t>x] \geq (n-1) \lim_{y\to\infty} E[t|t>y].$$

For the hyperexponential distribution,

$$E[t|t>z] = (a/u + (1-a)e^{-(v-u)z}/v)/(a+(1-a)e^{-(v-u)z})$$

so the condition becomes

$$(a/u + (1-a)e^{-(v-u)x}/v)/(a+(1-a)e^{-(v-u)x}) \geq 1/u - nC/(n-1)$$

or

$$av + (1-a)ue^{-(v-u)x} \geq (v - nCuv/(n-1))(a + (1-a)e^{-(v-u)x}).$$

Solving for x leads to

$$e^{(v-u)x} \geq ((1-a)(v - u - nCuv/(n-1)))/(nCauv/(n-1)).$$

Thus preemption is <u>not</u> desirable if

$$x > X_0 = \frac{1}{v-u} \log_e\left[\frac{(1-a)(v-u-nCuv/(n-1))}{nCauv/(n-1)}\right].$$

If $C \geq (n-1)(1-a)(v-u)/(nuv)$, then $X_0 < 0$ and preemption is never desirable with n requests in system. If $C \geq (1-a)(v-u)/(uv)$, preemption is never desirable no matter how many requests are in system. Figures 3 and 4 show how X_0 depends on the parameters a, u, v, and C.

III. AN UPPER BOUND ON THE VALUE OF PREEMPTION

Before attempting to characterize an optimal preemption strategy, we will derive a bound on how much preemption can possibly help. The bound is the average time in system under PPS with C=0. The bound must hold because PPS is optimal when C=0, and clearly, C>0 can only make average time in system increase. Under PPS, the expected total time in system in serving two requests can be calculated to be

$$T_{PPS} = 4\left[\frac{a}{u} + \frac{(1-a)}{v}\right] - \left[\frac{a^2}{u} + \frac{(1-a)^2}{v} + \frac{2a(1-a)(u^2+v^2)}{uv(u+v)}\right] .$$

With no preemption (equivalent to the first-come, first-served or FIFO discipline), the expected total time in system is

$$T_{FIFO} = 3\left[\frac{a}{u} + \frac{(1-a)}{v}\right]$$

since the mean of the hyperexponential distribution is a/u + (1-a)/v.

Using T_{PPS} and T_{FIFO}, we can calculate an upper bound on the percentage gain in average time in system attainable through preemption:

$$\frac{T_{FIFO}-T_{PPS}}{T_{FIFO}} = \frac{a(1-a)(1 - u/v)^2}{3(a + (1-a)u/v)(1 + u/v)} .$$

Figure 5 shows this bound on percentage gain as a function of a for various ratios of u to v.

IV. THE CHARACTER OF AN OPTIMAL STRATEGY

We will now characterize some properties of an optimal scheduling strategy for completing two requests, each unstarted initially. We were able to derive an expression for X_0, the point of attained service at which no further preemption is desirable. Similarly, there exist points X_1, X_2, X_3,... such that $X_0>X_1>X_2>X_3$... and once both requests have attained

service X_i, then there exists an optimal scheduling strategy for finishing them which will use at most i preemptions. Point X_i is defined as the least value of attained service for which the following statement holds: If the request not currently running has attained service x, then preemption is not justifiable until the attained service of the running request reaches at least X_{i-1}.

It is difficult to develop analytic expressions for the X_i, $i>0$, although they may be calculated for any particular set of parameter values, a, u, v, and C. If $X_k \leq 0 < X_{k-1}$, then each preemption up to the k^{th} will reduce the expected average time in system if it is used appropriately. However, any preemption beyond the k^{th} will improve the schedule only if some of the first k are not used to best advantage.

Because the median of a hyperexponential distribution is substantially less than its mean, we might expect that the marginal value of being allowed another preemption is a rapidly decreasing function of the number of preemptions. That is, if the difference between the best we can do with i preemptions and with i+1 preemptions is positive, then it is substantially less than the difference between the best we can do with i-1 preemptions and with i preemptions.

Table I demonstrates the validity of this conjecture for one specific set of parameter values. Note that most of the gain attainable through preemption can be obtained by using one preemption well.

A more obvious characteristic of an optimal strategy is that intervals of assignment will alternate between the two requests. A preemption will occur only when the attained service of the request currently being served exceeds the attained service of the other request by a sufficient amount

to justify delaying both requests by time C in order to preempt.

V. RULES TO BE COMPARED WITH THE OPTIMAL RULE

We continue our investigation of an optimal scheduling strategy by comparing the performance of known rules to the performance of an optimal strategy determined by a combinatorial search (OPT). Among the rules to be considered are PPS and FIFO, which were described earlier, and RR, CTSS, and DIFR. Round-robin (RR) scheduling involves serving the requests cyclically with fixed length quanta. The CTSS scheduling rule (used on the CTSS system at M.I.T. [7]) is a special case of the FB rule in which the quantum assigned at each queue level is double the quantum assigned at the next higher priority level.

The rule DIFR is intended to be the most obvious extension of the SR rule in which time lost in preemption is considered. For some fixed quantity, d, DIFR preempts the request being served if and only if its current rank, R(x), is at least d greater than the current rank of some other request. Clearly, d should be at least $D_0 = \lim_{y\to\infty} R(y) - R(X_0)$ since we know that preemption in favor of a request with attained service X_0 is not desirable. In this paper we will choose $d=D_0$.

For our hyperexponential distribution,

$$R(x) = \frac{ae^{-ux} + (1-a)e^{-vx}}{aue^{-ux} + (1-a)ve^{-vx}} .$$

Therefore,

$$D_0 = \lim_{y\to\infty} \left[\frac{ae^{-uy} + (1-a)e^{-vy}}{aue^{-uy} + (1-a)ve^{-vy}}\right] - \left[\frac{ae^{-uX_0} + (1-a)e^{-vX_0}}{aue^{-uX_0} + (1-a)ve^{-vX_0}}\right]$$

Using the equation for X_0 with n=2,

$$D_0 = \frac{1}{u} - \frac{a(v-u-2Cuv) + 2aCuv}{au(v-u-2Cuv) + v(2aCuv)}$$

or

$$D_0 = \frac{2Cv}{u(1+2Cv)} \quad .$$

For a given x, the DIFR rule must determine q(x) such that

$$R(q(x)) = R(x) + D_0 \ .$$

This is the point of attained service at which the request being served should be preempted in favor of a request with attained service x. Using the equations for rank and for D_0, then solving for q(x) yields

$$q(x) = \frac{1}{v-u} \log_e \left[\frac{K_1 e^{-(v-u)x} + K_2}{K_3 e^{-(v-u)x} + K_4} \right]$$

where $K_1 = 2C(1-a)v^3/(au)$

$K_2 = v-u + 2Cv(2v-u)$

$K_3 = v-u - 2Cuv$

and $K_4 = -2Cauv/(1-a)$.

In Figure 2, we observe that rank increases slowly at first, then more rapidly, and finally, it approaches a maximum value asymptotically. In general, we can locate the point of inflection in rank by setting the second derivative of rank equal to zero and solving for x. This yields

$$x = \frac{1}{v-u} \log_e \left[\frac{(1-a)v}{au} \right].$$

This point is greater than zero when $u/v < (1-a)/a$.

If $u/v \geq (1-a)/a$, successive quanta assigned by DIFR will be monotone increasing. Otherwise, the successive quanta will be of decreasing length out to the point of inflection, then the successive lengths will be increasing. Once X_0 is reached, DIFR assigns an infinite quantum. (Hyperexponential distributions composed from more than two

exponentials will generally lead to more complex patterns of interval lengths.)

Table II shows a comparison of the expected total time in system of serving two requests for each scheduling rule that we have discussed. The successive quanta assigned are also shown. We can make several observations. Every preemptive rule yields a significant decrease in total time in system relative to FIFO. By comparison with OPT, DIFR seems to overexagerate the variation in assignment quanta sizes. The RR rule appears significantly better than CTSS. This seems related to the fact that the first several quanta assigned by OPT are of nearly equal length.

These observations suggest yet another scheduling rule . The rule can be called "limited round-robin" (LIMRR). Requests are served cyclically, each receiving a fixed length quantum of service as under RR. However, when attained service reaches some threshold, the request is removed from the quantum allocation cycle, and placed in a lower priority queue from which the requests are completed in first-come, first served order. Such a discipline appears desirable since the optimal strategy assigns quanta of nearly constant length for a while, then the quanta rapidly approach infinity. The performance of LIMRR is included in Table II. Indeed, LIMRR does slightly better than RR, at least for the particular parameter values used.

VI. AN ESTIMATE OF THE NUMBER OF PREEMPTIONS NEEDED

We would like to be able to estimate the maximum number of preemptions used by OPT in serving two requests. Examination of several

sets of parameter values leads to the following rough approximation:

$$N_p = \frac{1}{2} \frac{R(\infty) - R(0)}{R(\infty) - R(X_0)}$$

$$= \frac{(1-a)(v-u)(1+2Cv)}{4Cv(au+(1-a)v)} .$$

For the parameter values of Table II, N_p is approximately 7.5, and OPT uses at most 7 preemptions.

VII. LESS RESTRICTED SITUATIONS

Most of what has been learned here with respect to two requests appears to extend to the case of n initially available requests. As witnessed in the calculation of X_0, the desirability of preemption at some point is not as sensitive to the number of requests in system as might be expected intuitively. Serving n requests instead of two would not significantly change the relative performance of the scheduling rules we have discussed.

However, consider a situation in which all requests need not be initially available. Then a new arrival may find one or more partially served requests in system. If a preemption is worthwhile, the new request should be served at least until the next arrival or until its attained service exceeds that of some other request.

The round-robin based rules, which do well when all requests are initially available, have a weakness when confronted with arrivals. They will sometimes cause preemption of one request in favor of a request with greater attained service time. With hyperexponentially distributed service times such a practice increases expected time in system. The round-robin strategy is dominated by a multi-level feedback rule which assigns identical quanta at each queue level. This rule performs identically

to RR in the absence of arrivals, but it will never preempt a request in favor of a request with greater attained service.

VIII. CONCLUSIONS

This investigation has disproven two intuitively appealing conjectures. We found that number of requests in system was less critical to the preemption decision than might be expected. We found that RR performs better than CTSS despite the fact that the quanta assigned by the optimal rule get longer and longer, and reach infinite length when attained service reaches X_0. The optimal scheduling rule resembles an FB rule, but how to determine the quantum length to be assigned at each queue level is unclear.

We have dealt specifically with only one special case of a hyperexponential distribution. However, similar conclusions hold for any service time distribution which has finite moments and the property that expected remaining service time increases with attained service time. This claim is based on the fact that every such distribution approaches an exponential distribution when conditioned on increasing values of attained service. We have dealt only with average time in system as a measure of performance. Many other performance criteria might also be considered. For example, requests with short service times might be thought to deserve better service than longer requests. If so, the optimal scheduling rule would tend to preempt with greater frequency than before.

The results presented here are not of immediate practical value. Their importance lies in an increased understanding of how the loss of processor time in preemption influences the performance of scheduling rules whose properties are known for the case in which time lost in preemption is negligible.

FIGURE CAPTIONS

Figure 1. A 2 phase hyperexponential distribution and an exponential distribution with the same mean for a=.1, u=.025, v=.250. Note that $w = uv/(av+(1-a)u)$.

Figure 2. The expected remaining service time and the rank of the hyperexponential distribution of Figure 1. The lower dotted line is the conditional mean and rank of $f(t) = we^{-wt}$. The upper dotted line is the conditional mean and rank of $f(t) = ue^{-ut}$.

Figure 3. The point beyond which no preemption is desirable versus C for 2 requests and n requests with a=.1, u=.025, v=.250.

Figure 4. The point beyond which no preemption is desirable for C=.125, v=.250.

Figure 5. The relative gain of preemptive processor sharing over first-come, first-served with C=0. These provide upper bounds on the potential gain of preemptive scheduling with C>0.

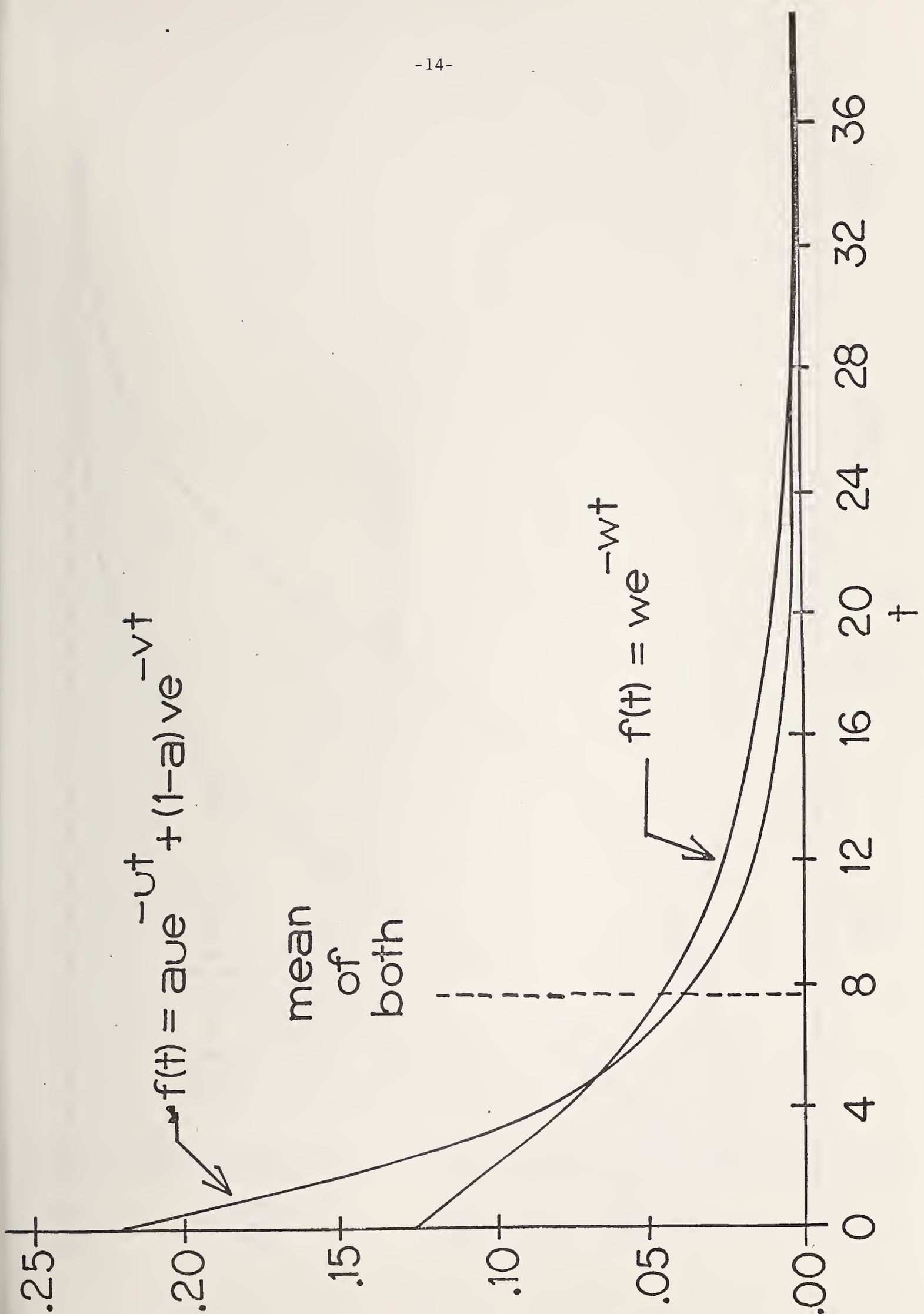
.25
.20
.15
.10
.05
.00
0
4
8
12
16
20
24
28
32
36
t
f(t) = aue^-ut + (1-a) ve^-vt
mean of both
f(t) = we^-wt

Figure 1

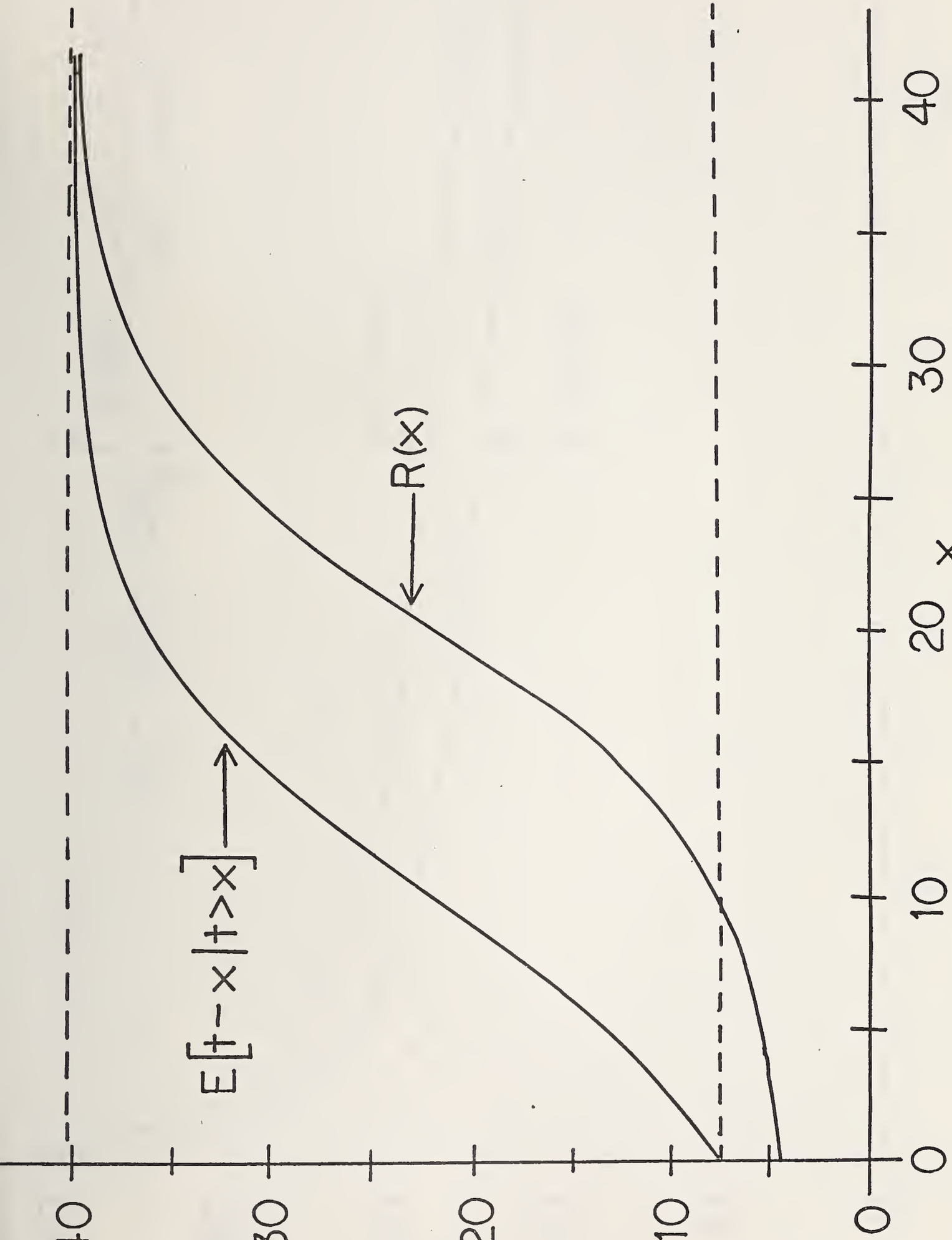

Figure 2

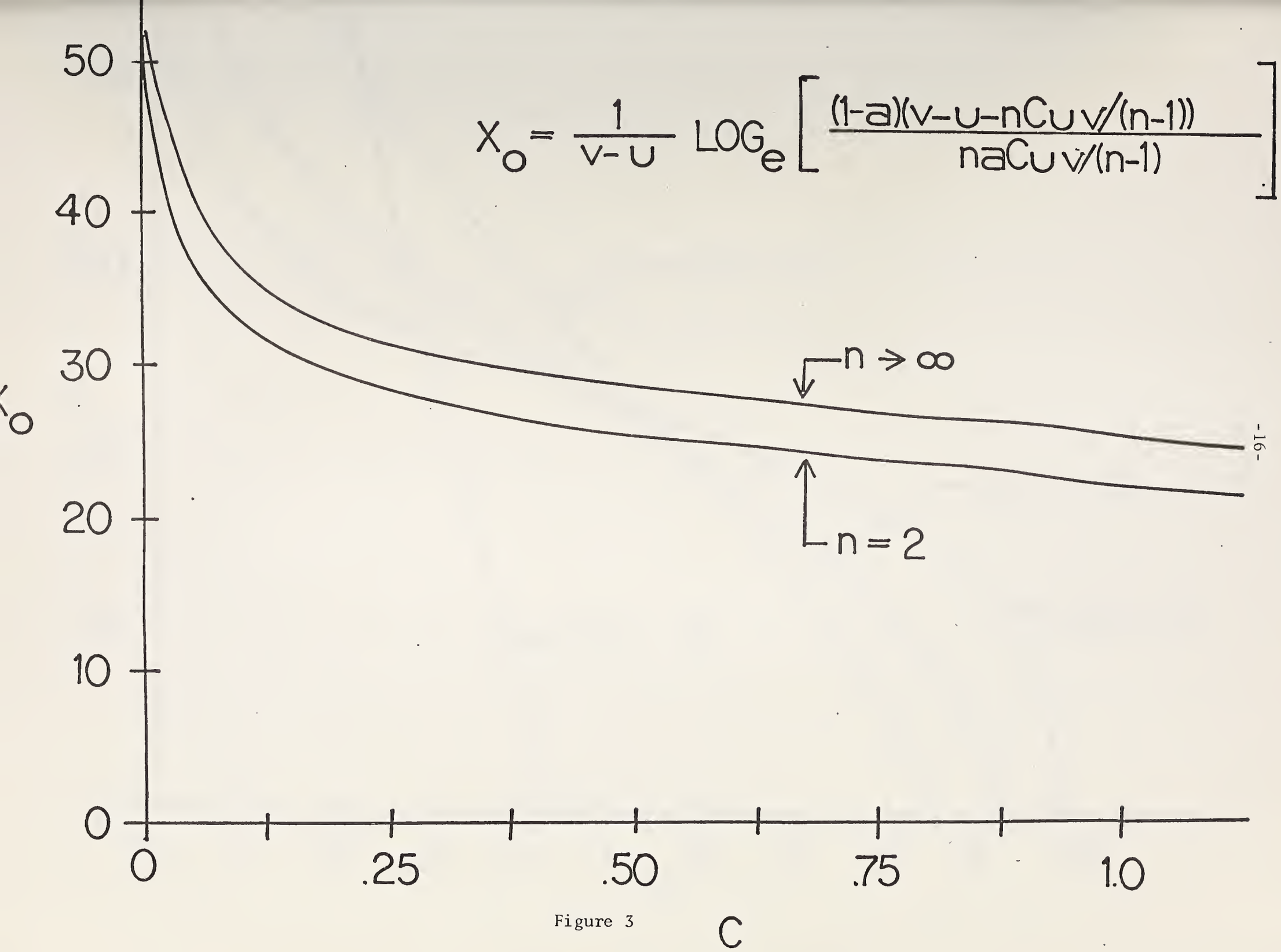
$X_o = \frac{1}{v-u} \text{LOG}_e \left[\frac{(1-a)(v-u-nCuv/(n-1))}{naCuv/(n-1)} \right]$
50
40
30
20
10
0
X_o
n → ∞
n = 2
0
.25
.50
.75
1.0
C

Figure 3

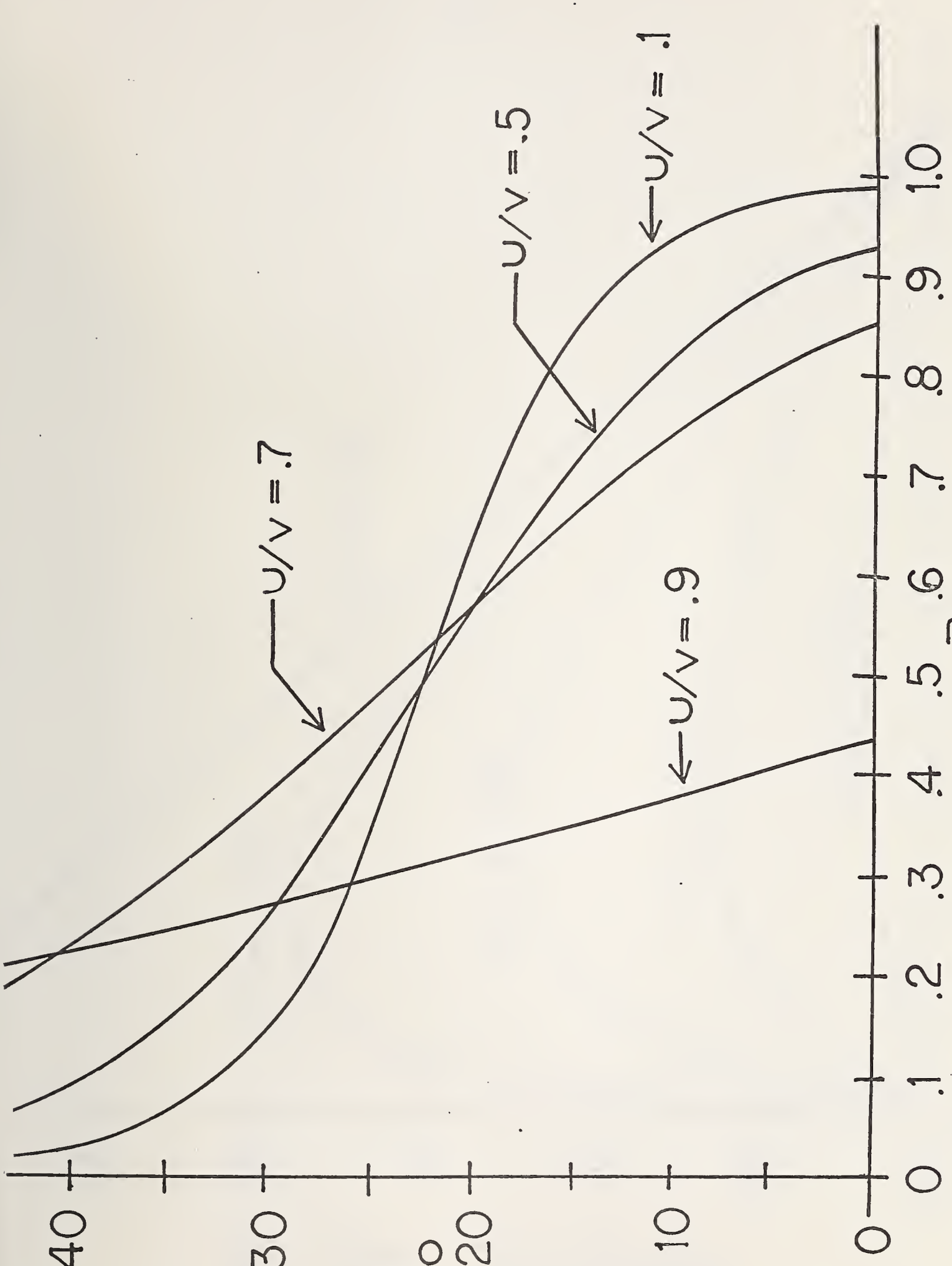

Figure 4

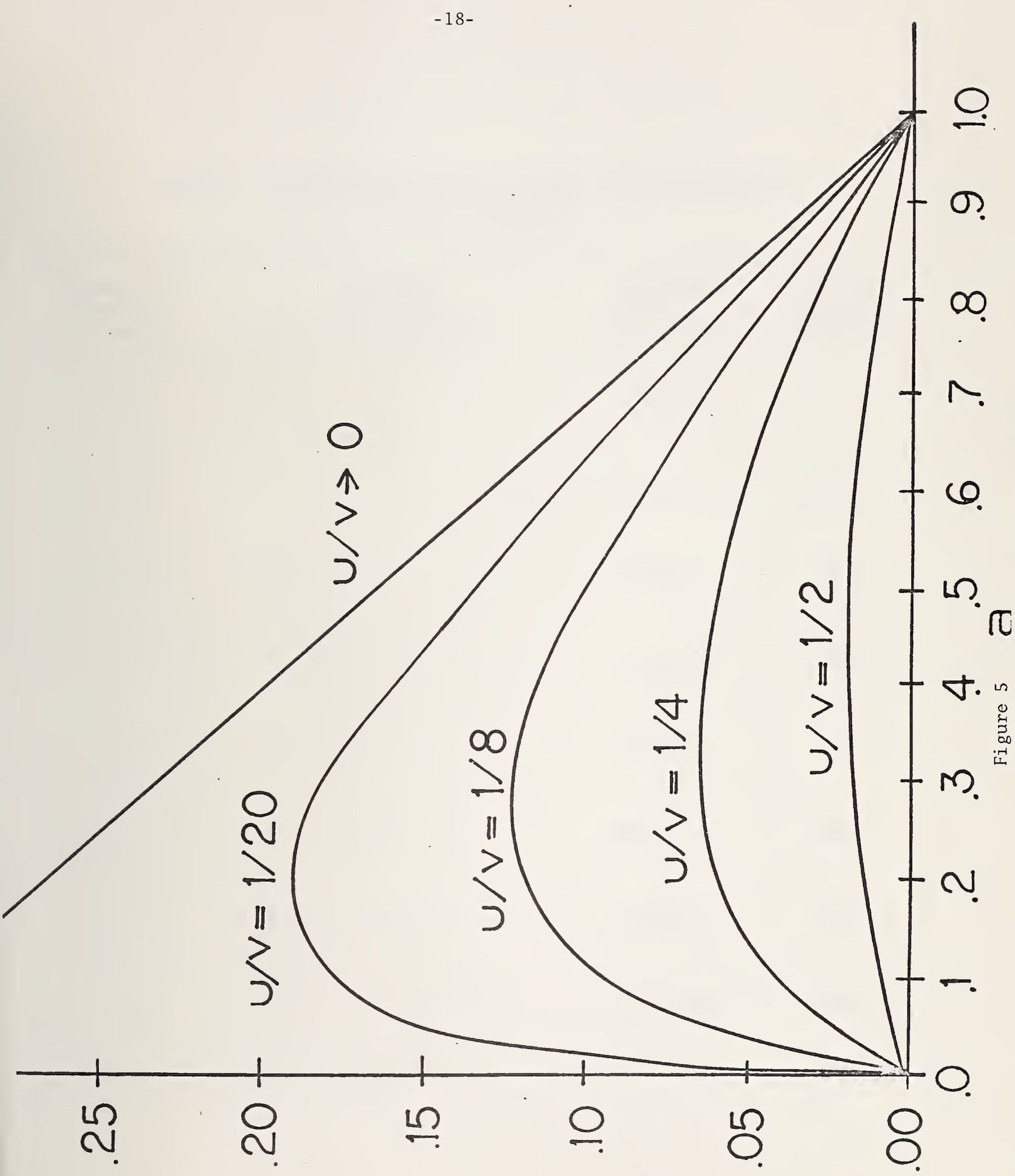
POTENTIAL GAIN %
.25
.20
.15
.10
.05
.00
0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
a
U/v = 1/20
U/v = 1/8
U/v = 1/4
U/v = 1/2
U/v → 0

Figure 5

TABLE I. Optimal Scheduling Strategies For Various Numbers Of Preemptions Allowed With n=2, C=1/8, a=1/10, u=1/40, And v=1/4

Number Of Preemptions Allowed	Assignment Intervals (as nearest integer)	Expected Total Waiting Time	Savings From Last Preemption
0	$\begin{matrix} \infty \\ \infty \end{matrix}$	22.8000	———
1	$\begin{matrix} 11 & \infty \\ \infty \end{matrix}$	20.5794	2.2206
2	$\begin{matrix} 6 & \infty \\ 16 & \infty \end{matrix}$	20.3784	.2010
3	$\begin{matrix} 6 & 16 & \infty \\ 12 & \infty \end{matrix}$	20.3363	.0421
4	$\begin{matrix} 5 & 11 & \infty \\ 10 & 16 & \infty \end{matrix}$	20.3249	.0114
5	$\begin{matrix} 5 & 8 & 17 & \infty \\ 9 & 10 & \infty \end{matrix}$	20.3215	.0034
6	$\begin{matrix} 5 & 8 & 11 & \infty \\ 9 & 9 & 17 & \infty \end{matrix}$	20.3210	.0005
7	$\begin{matrix} 5 & 8 & 9 & 21 & \infty \\ 9 & 8 & 12 & \infty \end{matrix}$	20.3208	.0002

TABLE II. Comparing Various Scheduling Strategies For n=2, C=1/8, a=1/10, u=1/40, And v=1/4

Scheduling Rules	Assignment Intervals (as nearest integer)	Expected Total Waiting Time
PPS	$e_1\ e_2\ e_3\ e_4\ e_5\ e_6\ e_7\ e_8$ --- where C=0 and e→0	20.1490
OPT	$5_1\ 9_2\ 8_3\ 8_4\ 9_5\ 12_6\ 21_7\ \infty_8\ \infty$	20.3208
LIMRR*	$6_1\ 6_2\ 6_3\ 6_4\ 6_5\ 6_6\ 6_7\ 6_8\ \infty\ \infty$	20.3549
RR*	$6_1\ 6_2\ 6_3\ 6_4\ 6_5\ 6_6\ 6_7\ 6_8$ ---	20.3567
CTSS*	$6_1\ 6_2\ 12_3\ 12_4\ 24_5\ 24_6\ 48_7\ 48_8$ ---	20.3911
DIFR	$9_1\ 12_2\ 5_3\ 4_4\ 3_5\ 2_6\ 2_7\ 2_8\ 2\ 3\ 4\ 4\ 5\ 13\ \infty\ \infty$	20.3946
FIFO	$\infty_1\ \infty_2$	22.8000

* An optimal quantum size is used.

REFERENCES

[1] F. Baskett, "Mathematical models of multiprogrammed computer systems," Ph.D. diss., Univ. of Texas at Austin, December 1970.

[2] R. Conway, W. Maxwell, and L. Miller, Theory Of Scheduling, Addison-Wesley, Palo Alto, 1967.

[3] K. Sevcik, "The use of service time distributions in scheduling," Ph.D. diss., Committee on Information Sciences, Univ. of Chicago, August 1971.

[4] E. G. Coffman, "Studying multiprogramming systems," Datamation, June 1967.

[5] L. E. Schrage, "The queue M/G/1 with feedback to lower priority queues," Management Science, Vol. 13, 1967, 466-474.

[6] M. Rothkopf, "Scheduling with random service times," Management Science, Vol. 12, 1966, 707-713.

[7] A. L. Scherr, An Analysis Of Time-shared Computer Systems, M.I.T. Press, Cambridge, Mass., 1967.